CSE 351 Section 9
Spring 2010

Virtual Memory

Overview

Announcements

- Lab 7 (Memory Allocation) due date pushed back to next Thursday, June 3rd
- HW 8 (Linking, Cache, Virtual Memory, Garbage Collection) due on Wednesday, June 2nd, at beginning of lecture.

Today's Agenda

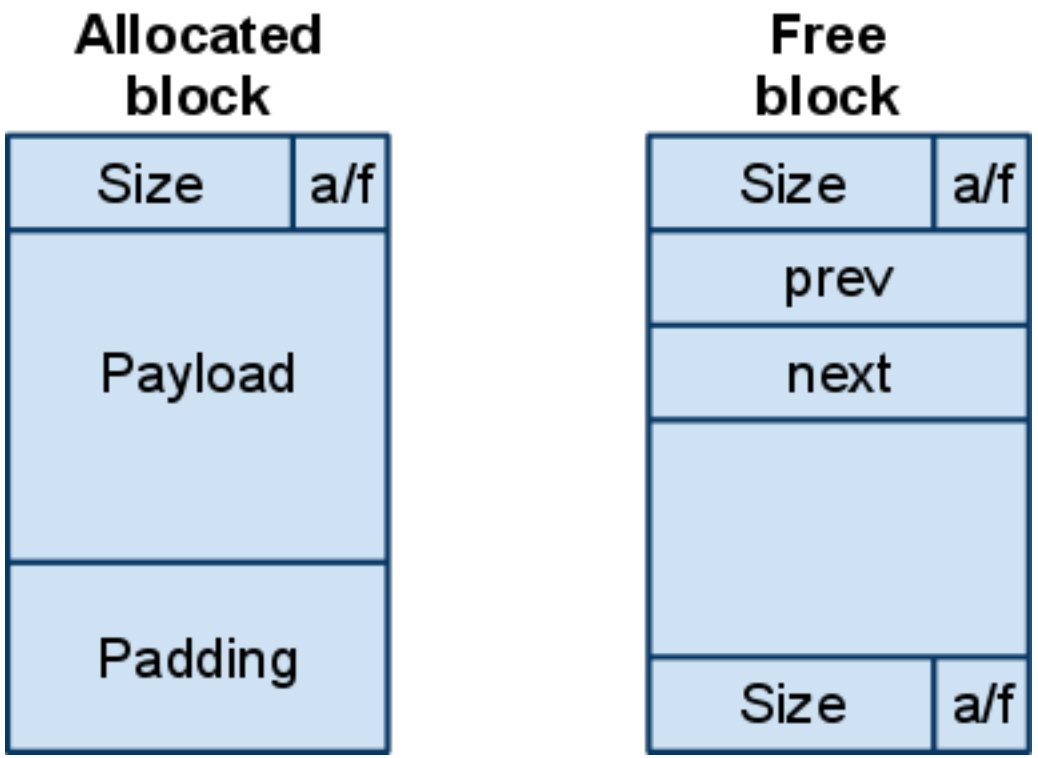
- Lab 7 questions
- Virtual Memory Review
  - Practice Problem 9.4 from book
- Garbage Collection

Additional Lab 7 Tips

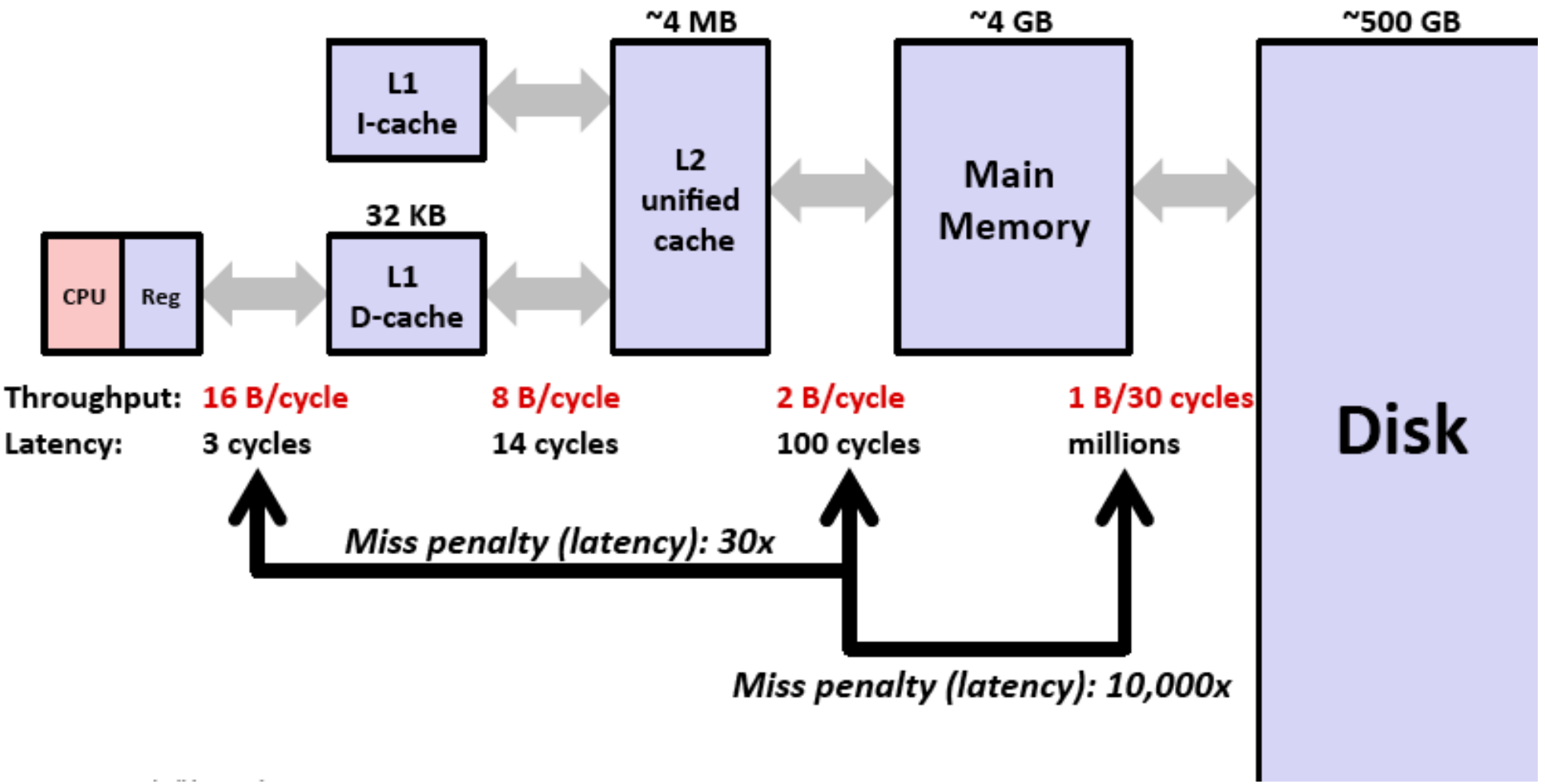
- Don't worry about splitting or utilization/performance at first, get it to pass `short1-bal.rep` first.
- Your performance scores will vary on attu, don't worry.
  - We'll use a controlled environment for grading.
- Enable debugging support in your mdriver program
  - Change "`CFLAGS= -g -Wall`" in Makefile
- You don't have to disassemble anymore.
  - You can step over whole lines of source code with step/next instead of stepi/nexti
- You need to update TAG_USED and TAG_PRECEDING_USED when you allocate and free.

Lab 7 Correction: BlockInfo

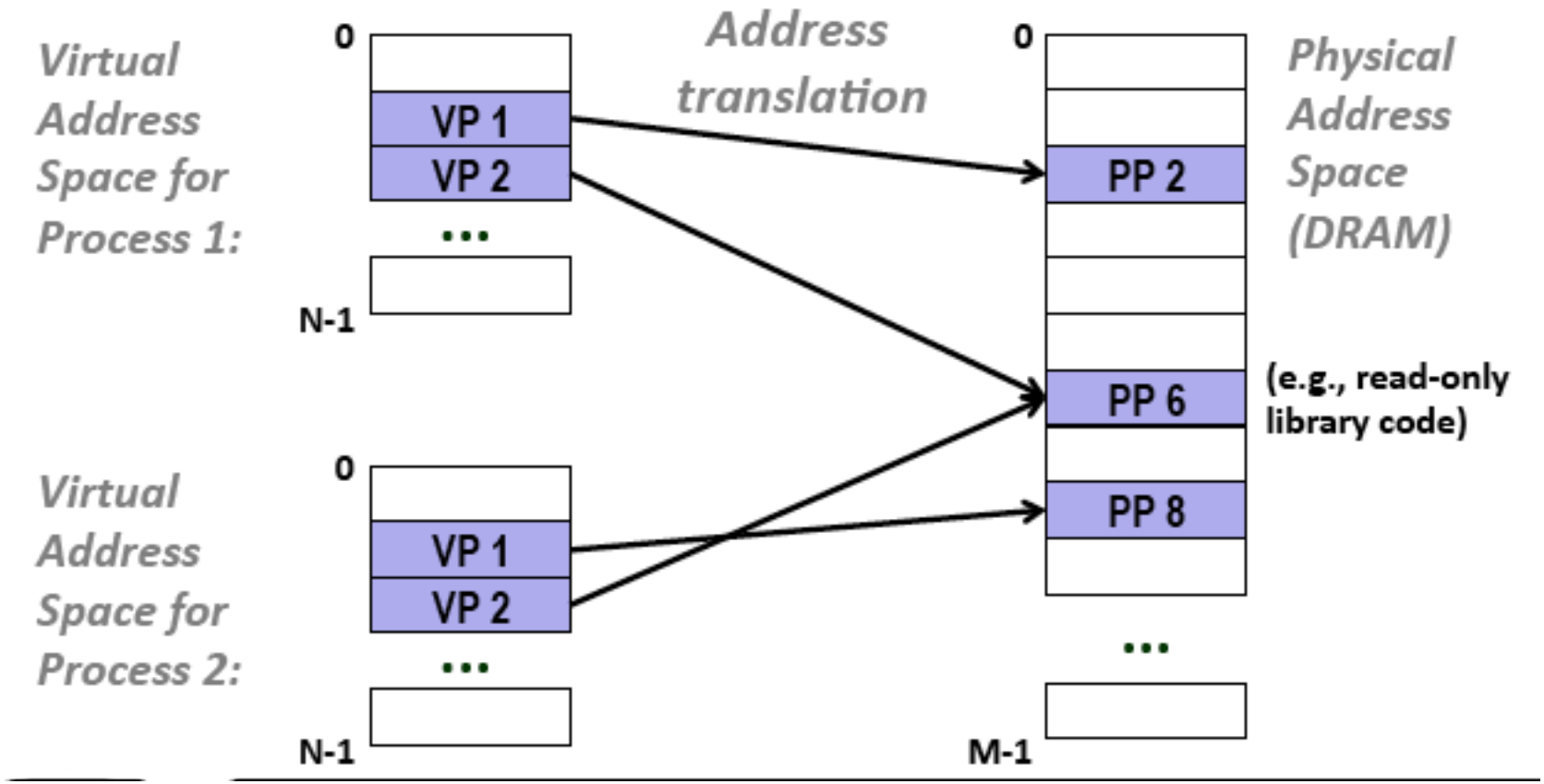
Allocated blocks don't necessarily have a footer (boundary tag)



Memory Hierarchy

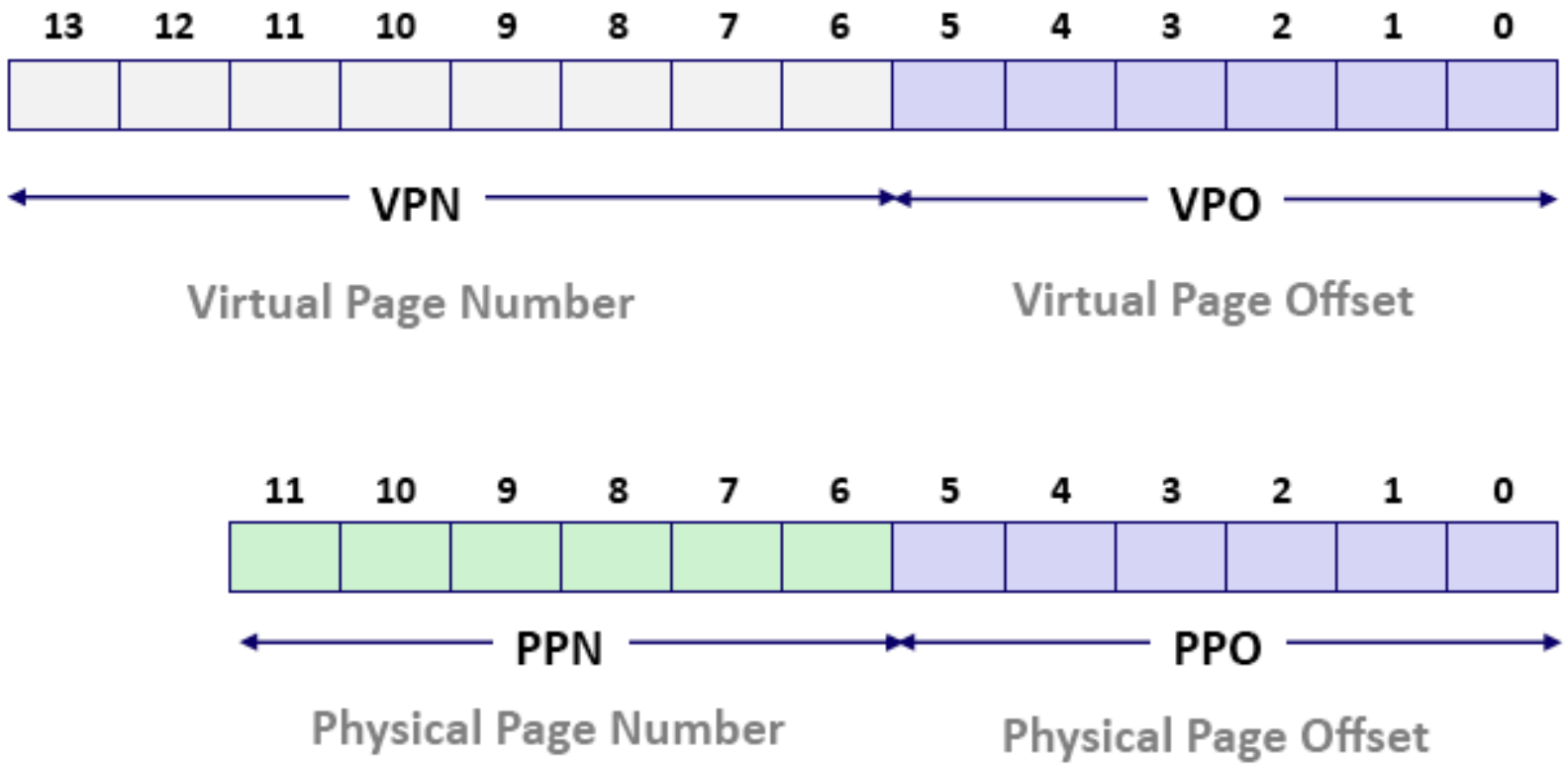


Virtual Memory: Protect & Manage

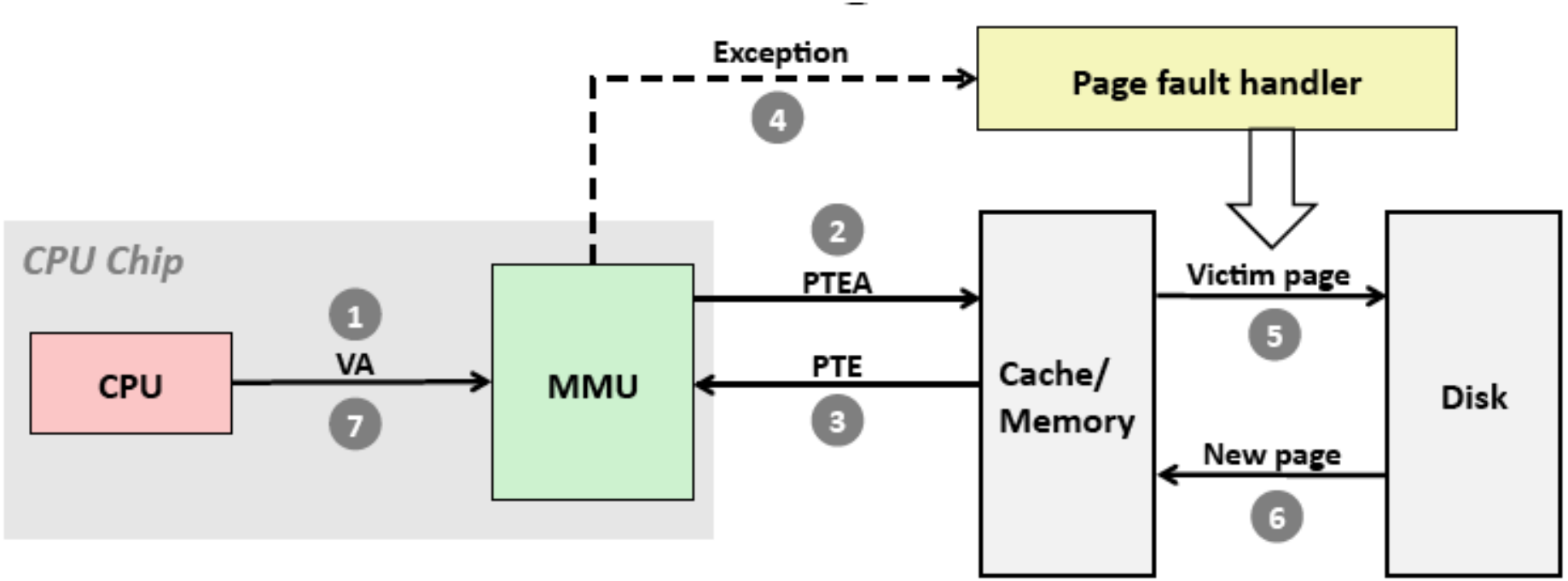


Virtual Memory: Addressing

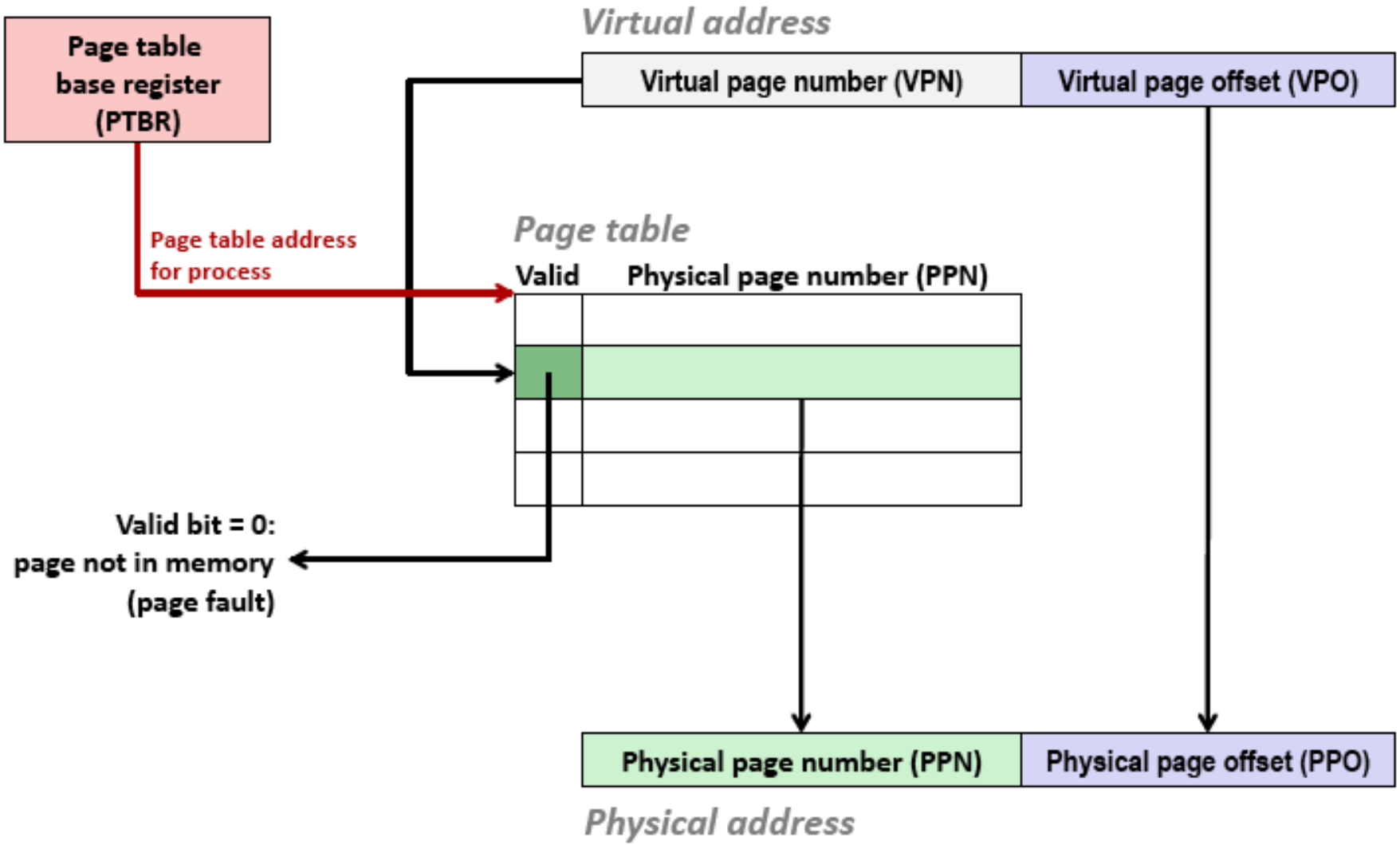
- **Addressing**
  - 14-bit virtual addresses
  - 12-bit physical address
  - Page size = 64 bytes



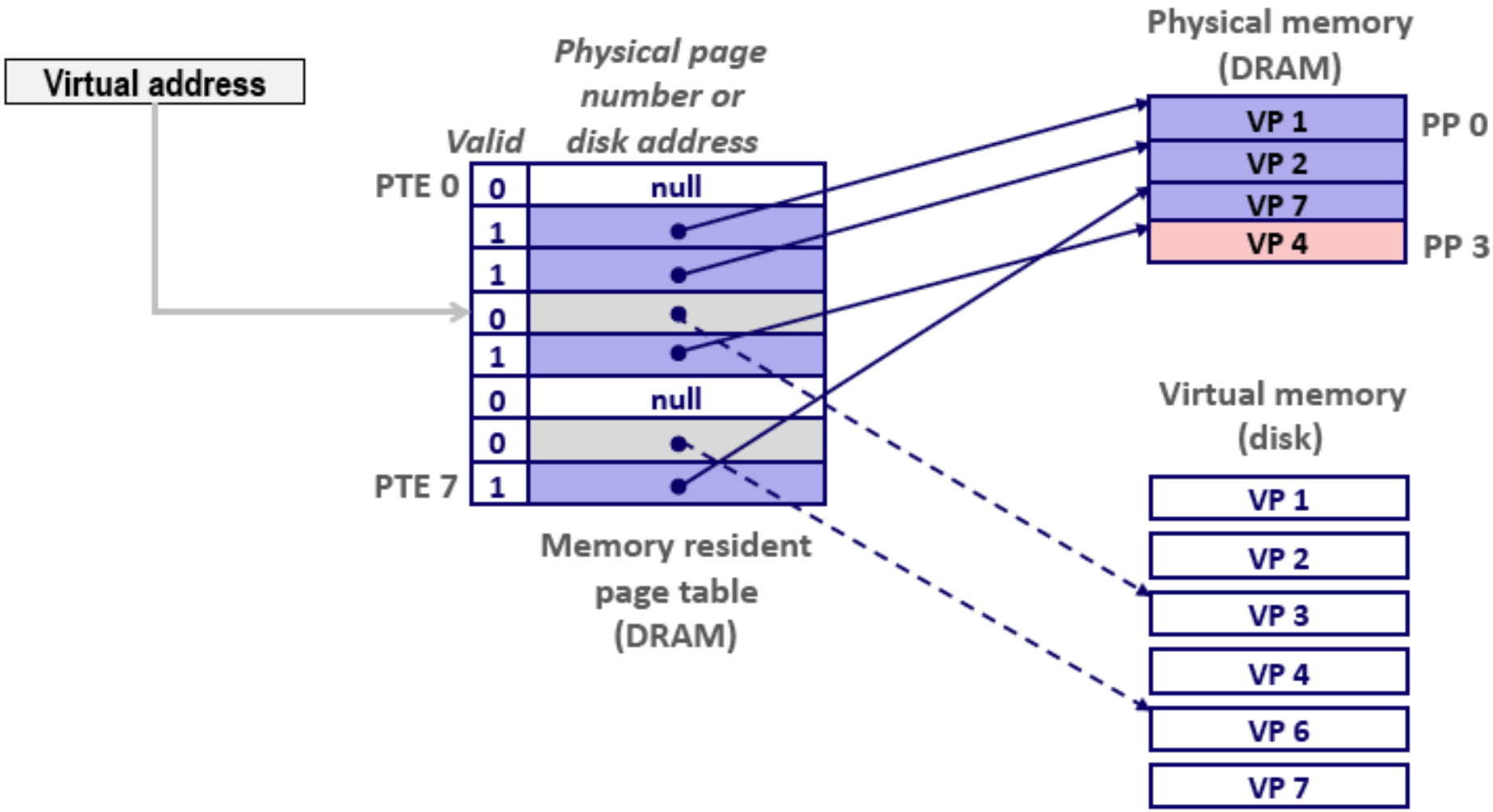
Virtual Memory: Pages and Faults



Virtual Memory: Page Table

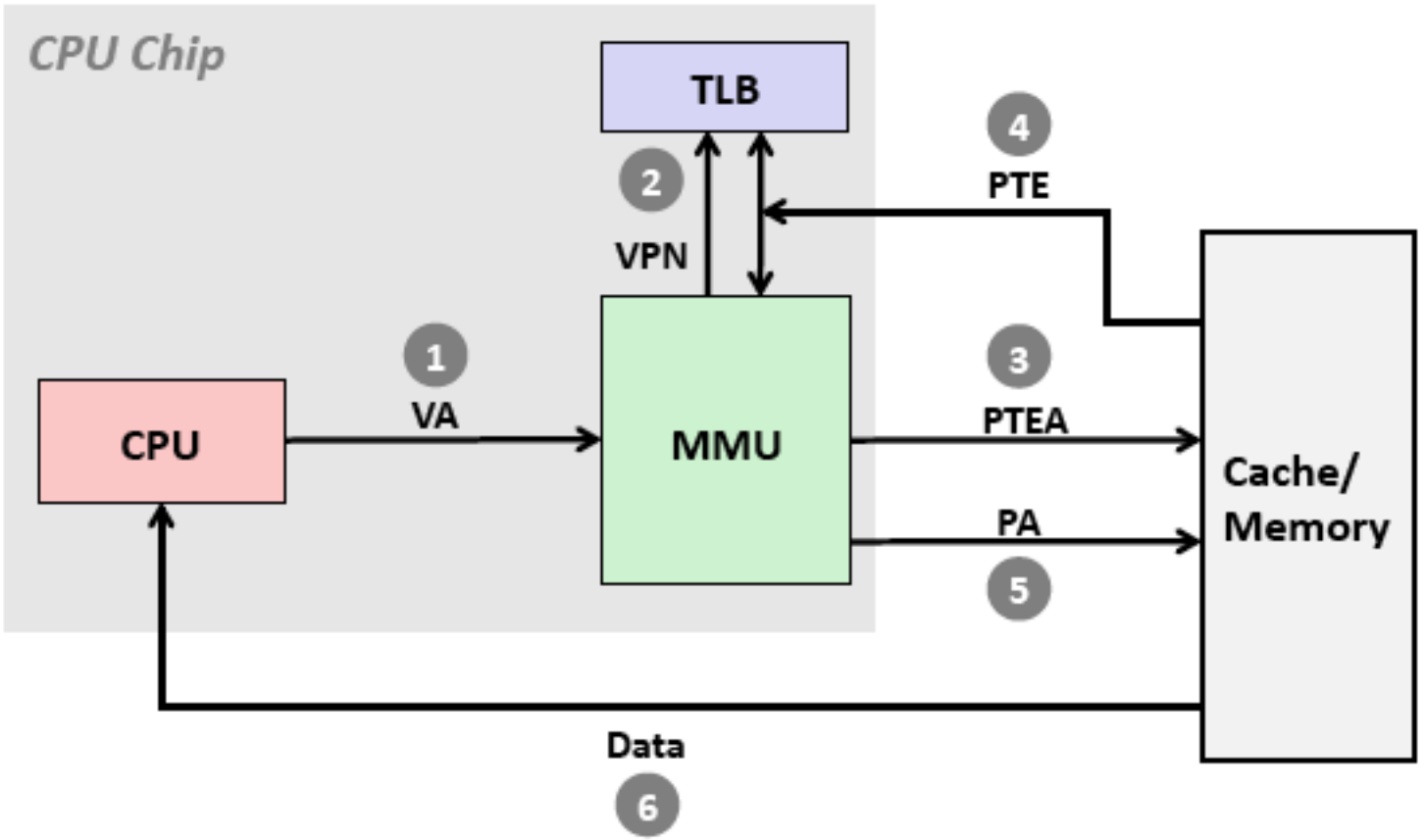


Virtual Memory: Page Table Entries



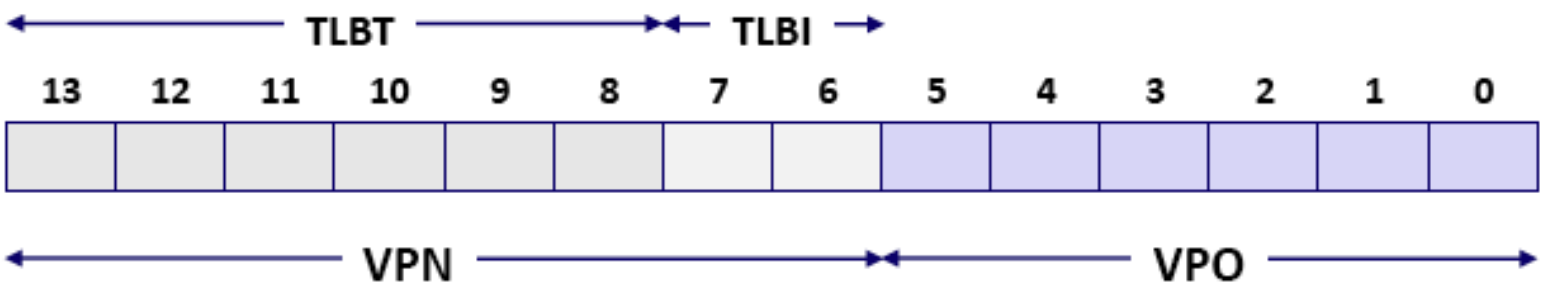
Virtual Memory: TLB

Translation lookaside buffer speeds up access to page table.



Virtual Memory: TLB Example

- 16 entries
- 4-way associative



Set	Tag	PPN	Valid	Tag	PPN	Valid	Tag	PPN	Valid	Tag	PPN	Valid
0	03	–	0	09	0D	1	00	–	0	07	02	1
1	03	2D	1	02	–	0	04	–	0	0A	–	0
2	02	–	0	08	–	0	06	–	0	03	–	0
3	07	–	0	03	0D	1	0A	34	1	02	–	0

Virtual Memory: Practice Problem 9.4

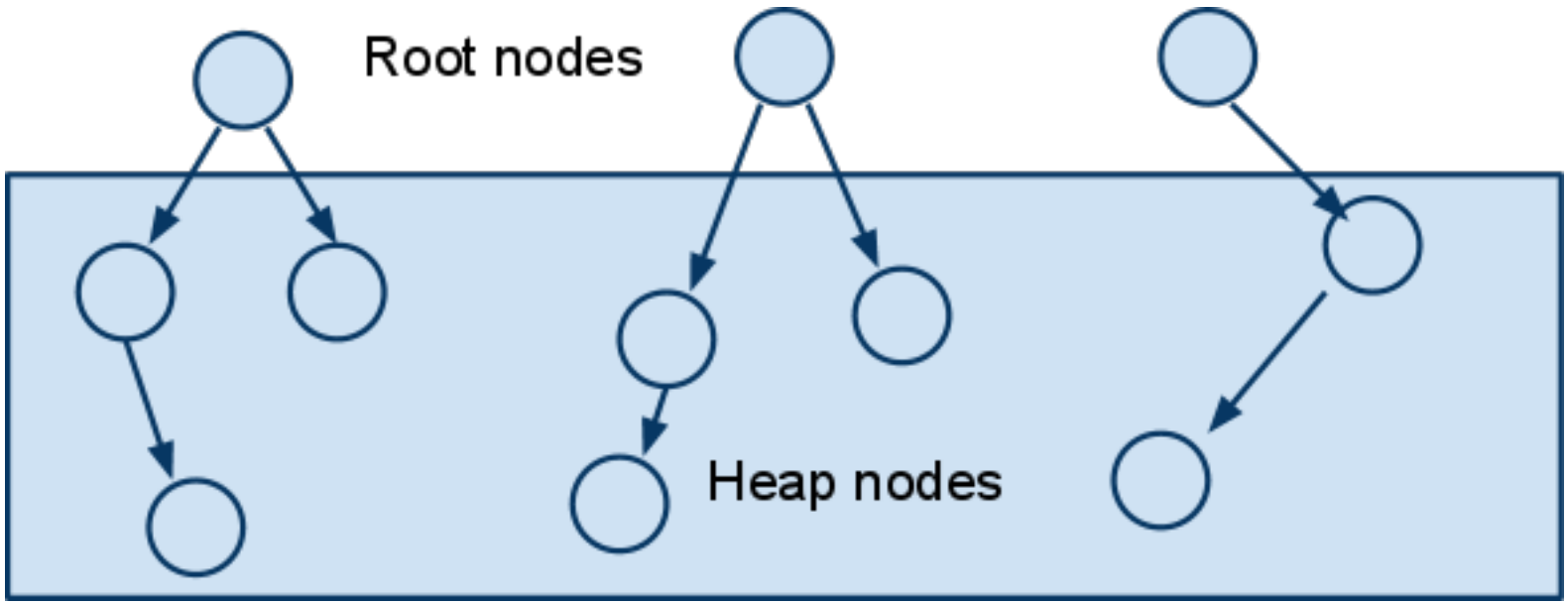
Page 798 in your textbook.
Using memory state on page 796, or handout from lecture.

Virtual Address: 0x3d7
A. Virtual address format? (14 bits)
B. Address translation: VPN, TLBI, TLBT, Hit?, Fault?, PPN
C. Physical address format? (12 bits)
D. Physical memory: BO, CI, CT, Hit?, Byte returned

Garbage Collection Review

Automatic memory management via garbage collection provides many benefits to programmers:

- no explicit memory leaks
- no dangling or double-freed pointers



Garbage Collecting Analogy

Imagine that you share a fridge with your roommates. Asking people to label their food and throwing out unlabeled food is similar to the **mark-and-sweep algorithm**.

Garbage Collecting Puzzle 1

1. How can we still leak memory in a garbage-collected language like Java?

Garbage Collecting Puzzle 2

2. Suppose we keep a reference counter with every object.

We increment the counter every time a new pointer references the object.
We decrement the counter every time an existing pointer goes away (either from stack frame or garbage collection).

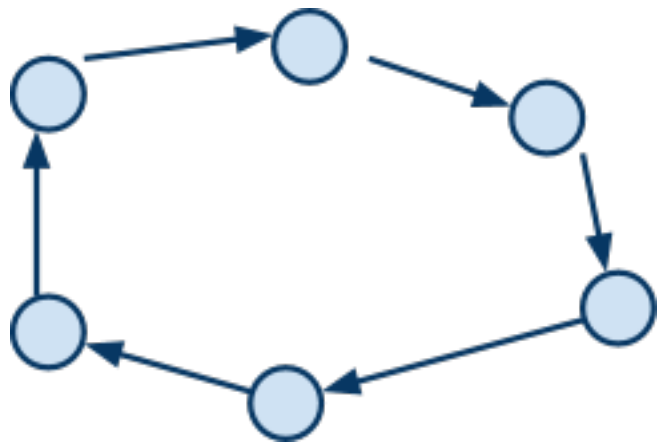
When a reference counter reaches zero, we free the object.

Will this garbage collection algorithm work?

Garbage Collecting Solutions

1. We can still write our programs poorly which keep lots of global variables or long-lived data structures around.

2. Reference counting won't detect a pointer cycle.



Good luck!

Lab 7: due next Thursday, 11:59pm

Homework 8: due next Wednesday, beginning of lecture

E-mail cse351-tas@cs if you have questions